This application is submitted in the name of Brian Lo Bue, Darrell Shively and Larry Nadeau, assignors to Cisco Technology, Inc., a California corporation.

SPECIFICATION

TITLE OF THE INVENTION

ACTIVE CALL CONTEXT RECONSTRUCTION FOR PRIMARY/BACKUP RESOURCE MANAGER SERVERS

FIELD OF THE INVENTION

The present invention relates to a method and apparatus to reconstruct an interrupted user's network connection from an updated data string held in storage by the network server.

BACKGROUND OF THE INVENTION

Computer networks have developed into an integral part of society and the economy. Network users obtain access from their host computer with dial-up capability to another computer with which to communicate through a variety of systems currently available. A user transmits a request or call from an access point across a medium to the remote site. The call will be identified by selected attributes in an access request data packet that may be used by the network accounting management. To facilitate access by authorized call-in users and network subscribers, a variety of network servers have been developed that may include computer hardware, software and/or firmware. These servers are frequently designed to handle specific tasks within the network and operate with a dedicated database. A call-in user may thus be dependent on more than one server for continued access to the network, and if one server fails while others continue to operate, the user may nonetheless be disconnected from the network and be forced to establish a new connection even though the network remains active.

The time consumed from repeated interruptions can lead to diminished productivity and severe frustration if critical data are denied or corrupted due to a recalcitrant server. Consequently, network administrators have sought to address this and other concerns by ensuring high reliability for servers and establishing backup systems. However, a backup server without access to the call's status from the identification in the data packet may be unable to maintain a previously established call, resulting in the call being cut off. Consequently, aside from efforts to ensure high reliability of servers, the industry also requires a backup system with the ability to hand-off a call-in user's network connection from a server that has failed to a backup server by ensuring that call identification information is received by a backup server in a timely fashion.

A connection from an access point to a network at a point of presence (PoP) may be maintained by an internet service provider (ISP) or a telephone company using communications media such as a public switched telephone network (PSTN), integrated services digital network (ISDN), or a cable television system, using one of several available mechanisms or protocols. Such protocols include the decentralized Institute of Electrical & Electronic Engineers (IEEE) standard 802.3 called Ethernet™, the token ring IEEE standard 802.5 incorporating a special bit-pattern to control transmission order, the asynchronous digital subscriber line (ADSL) under the American National

Standards Institute (ANSI) T1.413 standard, the hybrid fiber coax (HFC) used by cable television providers, or others as is well known in the art. An access point may contain a variety of server types for particular functions. These types include the authentication, authorization and accounting (AAA) server, the network access server (NAS), the resource pool manager server (RPMS), the home gateway router (HGR), the digital subscriber line aggregation multiplexer (DSLAM), along with many others well known in the art.

A call-in user seeking a connection to the network may place a call across telephone lines or other media to a NAS through a particular port of the NAS, such as a modem port or ISDN port. The NAS answers the call, becoming coupled to the user, and sends the call type and dialed number information service (DNIS) information to the RPMS, which matches the combination to a call discrimination table and compares the network resources available to the session counts. Call types include speech, digital and others known in the art. The call is rejected if the call type–DNIS combination appears in the call discrimination table. If the customer profile session limits exceed threshold values, the call may be rejected or assigned a busy signal. If the call is accepted, it is assigned to the NAS that answered it. A RPMS may provide resource management, dial services and call discrimination for a regional PoP or for a NAS connected to multiple ports.

A RPMS enables telephone companies and ISPs to count, control, manage and provide accounting data on shared resources for wholesale virtual private dial-up network (VPDN) and retail dial network services across one or more NAS stacks. By tracking threshold access limits, the RPMS verifies to the NAS that there exist sufficient network resources to enable a user calling in to become connected to the network (provided that the user has authorization). An illustration of the logic used by a RPMS can be seen in the flow diagram 10 of FIG. 1. An input 12 containing call type–DNIS information is provided to a call discriminator query 14, which compares the information to a series of discriminators implemented as a call discrimination table. If the call matches the table, it is unauthorized and the call is treated to rejection 16. If no match is found, the DNIS customer profile is queried 18, and if none is found, a default profile is queried 20. If no default match is found, the call may be rejected 22. If the DNIS customer profile exists after query 18, the connection threshold is queried 24, and if not reached, or if a default customer match is found in the default query 20, then the number of resources is queried 26. If sufficient resources are available, the call may be answered 28, whereupon the VPDN group is verified 30.

If the call does not match the VPDN group, a first retail query is performed 32, and if refused, the call is rejected 34. If retail is accepted, the call is processed at retail cost 36. If the call matches the VPDN group, the domain name in the DNIS is queried 38. If the domain name does not match, a second retail query is performed 40. If refused, the call is rejected 42, and if accepted the call is processed at retail cost 44. If the domain name query 38 matches, the session and overflow thresholds are queried 46. If the thresholds are exceeded, the call is rejected 48, and if not a tunnel is negotiated 50.

Returning to the maximum connections query 24, if the connection threshold has been reached, the overflow availability is queried 52, and if exceeded, the call may be rejected 54. If availability exists, the availability of resources is queried 56. If the resources available queries 26 or 56 are negative, the call may be rejected 58. If resources are available, the call is answered 60, with continued procedures to the VPDN group verification query 30, and so forth.

The RPMS enables shared resources to be used across multiple NASes for various resource allocation schemes (performing session counting on a group level). For example, NAS resource groups may be combined with different modem services and call types (such as speech or digital) into resource data assignments. Resource groups may be configured on the NAS and assigned by the RPMS based on customer requirements. The RPMS may use resource management protocol (RMP) software to communicate with the NAS. An illustration of this arrangement is illustrated in FIG. 2A, in which a NAS 62 is featured with RMP installed and connected to a RPMS 64 via a RMP interface 66. The call type–DNIS information transmitted to the RPMS 64 and the approval or rejection response received by the NAS 30 are transferred through the RMP interface 66 using the RMP protocol 68.

The RPMS may be composed of a server platform with appropriate RMP software, along with a Distributed Session Manager (DSM) library installed and linked to the server platform. A RPMS may be a scalable performance architecture (SPARC) hardware platform equipped with DSM software and connected to a database in a memory device physically distinct from the RPMS. The DSM represents a linked library to the RPMS to keep accounting data records for the RPMS, and it maintains session states across multiple servers. The database may hold the customer profiles, system configurations and other desired instrumentation.

A local AAA server may be used in a network architecture incorporating a RPMS for the purpose of tracking users that access the network through calling line identification (CLID) and for creating records of the accounting data related to such users from outside the data communications network. The RPMS may communicate with a local AAA server using an internet authentication protocol, such as Terminal

Access Controller Access Control System Plus (TACACS+) or Remote Access Dial-In User Service (RADIUS). In the absence of a local AAA server, the RPMS may provide only DNIS-based wholesale VPDN dial services under the domain name, and a remote AAA server may be used for user call tracking and management. When the call ends, the record for billing purposes may be sent to a report manager server.

As calls are received, the primary RPMS checks session counts to perform session management. These local counts are sent to the backup RPMS for synchronization. When the session counts approach a session limit, the primary RPMS reverses the exchange to get the session count from the backup RPMS for each call, thereby ensuring that an accurate session count is maintained and prevents more users from accessing the network than are permitted, a condition called “over-subscription”. However, this exchange may reduce performance when the customer profile approaches its session limit for resource allocation.

A client NAS may be configured with a list of RPMSes from which to attempt contacting a server on the fail-over list. The message data may be exchanged between the NAS and the server by the RMP. If the NAS cannot reach the first server on the list, it tries to contact the next server, and so forth. In a typical configuration, the primary RPMS would be first on the list, and the backup RPMS would be the second on the list, with no third server listed. An illustration of this list’s use by multiple NASes is depicted in FIG. 2B in which a wide area network (WAN) 70 is connected to a first RPMS labeled “A” 72, a second RPMS labeled “B” 74, both independent of each other, and a backup RPMS labeled “C” 76. A first NAS stack 78 with a first server list for RPMSes “A” and “C” is connected to RPMS “A” 72. A second NAS stack 80 with a second list for RPMSes “B” and “C” is connected to RPMS “B” 74. The NAS stacks 78 and 80 are connected to their respective local RPMSes "A" 72 and "B" 74, respectively through RMP 82, and via the WAN 70 to RPMS "C" 76. If RPMS "A" 72 fails, the first list on the first NAS stack 78 would "roll over" or transfer resource management to RPMS "C" 76. Similarly, if RPMS "B" 74 fails, the second list on the NAS stack 80 would likewise proceed to RPMS "C" 76. However, if the backup RPMS lacks the information for call reconstruction, current calls may be discontinued.

In the event that either the primary RPMS or its database is unreachable, the call may be interrupted while the NAS initiates a timing switch. If the call is not restored, the switch times out, causing the call to be dropped from the NAS and a busy signal sent to the call-in user. Incorporation of a backup RPMS might initiate new user calls begun subsequent to the primary RPMS access failure, but absent a mechanism to restore the interrupted calls, the continuity of service for previous users would not be feasible. To avoid interruption of a call, the primary and backup servers must be in communication with each other and share information about a call's context as it is updated, thereby consuming valuable communication bandwidth.

SUMMARY OF THE INVENTION

A method and apparatus for network server recovery maintains an ongoing call by reconstructing its call context from a server-state attribute (SSA) that is generated by a first server, recorded in separate storage and retrieved by a second server in the event that a connection to the first server is interrupted. The SSA encodes call data that can be used to enable a server to maintain the call. The separate storage is preferably associated with a network access server (NAS) that does not itself use the SSA.

BRIEF DESCRIPTION OF THE FIGURES

FIG. 1 is a flowchart illustrating the conventional RPMS logic for responding to a call.

FIG. 2A is a diagram illustrating a conventional connection between a NAS and a RPMS.

FIG. 2B is a diagram illustrating the maintenance of server lists in conjunction with primary RPMS systems and a backup RPMS in accordance with the prior art.

FIG. 3 is a diagram illustrating a network having a primary RPMS and a backup RPMS in accordance with a presently preferred embodiment of the present invention.

FIG. 4 is a diagram illustrating an example server-state attribute data string in accordance with a presently preferred embodiment of the present invention.

FIG. 5 is a process flow diagram illustrating the reconstruction of a data string in accordance with a presently preferred embodiment of the present invention.

FIG. 6 is a process flow diagram illustrating server failure response in accordance with a presently preferred embodiment of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Those of ordinary skill in the art will realize that the following description of the present invention is illustrative only and not in any way limiting. Other embodiments of the invention will readily suggest themselves to such skilled persons having the benefit of the within disclosure.

In accordance with a presently preferred embodiment of the present invention, the components, process steps, and/or data structures may be implemented using various types of operating systems, computing platforms, computer programs, and/or general purpose machines. In addition, those of ordinary skill in the art will readily recognize that devices of a less general purpose nature, such as hardwired devices, or the like, may also be used without departing from the scope and spirit of the inventive concepts disclosed herein.

The present invention relates to a method and apparatus to enable a backup server to reconstruct the context of an active call from a network call-in user connected to the network server after the connection to the primary server and/or its database has been severed or interrupted. Simply stated, this context may be restored from an updated data string constructed by the primary server and held in storage by a memory associated with the network server. This memory may be a separate memory device connected to and used by the network server, or may be a memory cache within the network server during its operation. In the within disclosure, the primary server may be represented by a local RPMS, the secondary server may be represented by a backup RPMS, and the network server may be represented by a NAS.

The data string may be constructed as a coded message from the primary server to the network server that the latter may store in associated memory but not need. This data string may include accounting information and may be updated periodically. When the network connection to the primary server has been unexpectedly severed from the network user as in a server or database failure (either of which may be considered a local server failure for the purposes of this invention), the backup server may query the network server to provide the data string from its associated memory, reconstruct the relevant information for the active calls before the timing switch signals a time-out and disconnects the user from the network.

In accordance with an embodiment of the invention, a redundant backup server processes messages for a "call" that began its processing on a primary server. A single RPMS backup server may support multiple local RPMSes. The RPMS backup server may provide a backup configuration and backup counters in case the local RPMS server becomes unavailable. To avoid interruption of a call, the data string transfer from the primary server to the backup server must occur in a manner transparent to the call-in user when a client NAS determines that the primary server is unreachable and switches or "rolls-over" to communicate with the backup server.

Since the NAS maintains data for each active call on its ports, an information packet constituting an "opaque" value may be stored that represents a "don't care" register with respect to the NAS, along with its own call data to make available to a server as needed. The opaque value, called the "server-state" attribute (SSA), contains context data that the server generates and updates for each active call, but may remain unused by the NAS. Since the service-state attribute may be produced from an aggregation of data elements in a specific sequence, this construction of the information packet called the SSA may be described as an encoding or "deparsing" procedure. The SSA value may contain particular call data that the server assembles from its call attribute table to describe the call and may be sent back to a memory separate from the server with each response message to the NAS. In the preferred embodiment, that memory may be associated with (that is, connected to) the NAS. The NAS updates its call attribute table to store the information packet referred to as the SSA. Hence, the SSA stored in the NAS acts as a tether to the information a backup server requires to continue processing a call without the need for another server to continuously exchange data with the backup server, thereby saving communication bandwidth.

When the NAS needs to communicate with the server concerning a particular active call, it may send the SSA along with the message, or it may provide the SSA if a server requests it. The contents of the SSA may be proprietary to the server and may be composed, in essence, of a condensed snapshot of the server's active call data as the call moves through its states until the call is terminated. The data fields containing the call data parameters may be separated by a particular character, such as commas, for delimiting fields. Other characters besides commas may be used as delimiters, as is trivially known to those with ordinary skill in the art.

If a server (such as a RPMS) receives a message from a memory storage such as a NAS for an active call that the server lacks in its call attribute tables, that server can obtain the SSA from the NAS. The server can then parse the SSA to "reconstruct" the context of the call in its call attribute tables and subsequently provide a response to the NAS. An example of this circumstance might occur if the NAS rolls over to communicate with the backup server from a failed primary server. The backup server would query the NAS for the SSA of a call originated during the operation of the primary server. After the call context reconstruction, the backup server provides the response to the NAS originally intended to be supplied by the primary server. The active call data continue to be available for billing and reports (such as for a local AAA server or in support of VPDN dial services), so that a meaningful call detail record may be generated when the call closes. Session and resource counts may be maintained by the backup server and restored to the primary server when the primary server becomes available.

This procedure reduces network traffic and increases system efficiency since the primary server need not feed a separate message to the backup server for each change in call context. With the connection between the servers and the NAS already established, the SSA may be embedded in messages that are being passed from the NAS and server for their normal business so the overhead of primary/backup communication, such as generating headers, opening connections, etc., may be completely eliminated. The NAS may be used as the instrument of storage for the server's call data in the event of a local server failure and only contains the latest data set for a call that is open. Otherwise, a more elaborate scheme would have to be devised between the primary and backup servers to ensure that active call data are current (meaning updated regularly) and free of "zombie" sessions, meaning inactive calls without a termination message.

Both primary and secondary databases may hold the customer profiles and system configurations. A RPMS may utilize database replication to ensure that databases running on geographically distributed primary RPMS installations contain the same customer profile information.

FIG. 3 illustrates an example configuration of a hypothetical redundant configuration incorporating the present invention. The network depicts an outside world 110 and the server group 112 sharing a telephone demarcation line 114. A call-in user or customer 116 on an ISP may be connected through a PSTN 118 in the outside world 110 across the demarcation line 114 to a NAS 120 in the server group 112. The NAS 120 may be connected to an Ethernet 122 that accesses other servers. While only a single NAS 120 is depicted in FIG. 3, more than one NAS 120 can be accommodated by the RPMS system in the server group 112. The primary or local RPMS 124 may be represented by a computer workstation 126 with a master DSM library 128, connected to the Ethernet 122 by means of a first port line 130. A primary database 132 may be dedicated to the primary or local RPMS 124 and be connected to the Ethernet 122 by a second port line 134. In this example configuration, only a single primary or local RPMS 124 is exhibited rather than several.

A secondary or backup RPMS 136 may be represented by a workstation 138 with a slave DSM library 140, connected to the Ethernet 122 by means of a third port line 142. A secondary database 144 may be dedicated to the secondary or backup RPMS 136 and be connected to the Ethernet 122 by a second port line 146.

If a primary or local RPMS 124 fails, loses its network connection 130 or is cut off from access to its master DSM library 128, the primary or local RPMS 124 will be isolated. With no information going in or out of the primary or local RPMS 124, the switch will timeout pending transactions. The administrator will need to re-establish connectivity for the ISP customer 116 in short order. In the meantime, the NAS 120 will use the secondary or backup RPMS 136 for authorizations and accounting. When connectivity is restored, the secondary or backup RPMS 136 will attempt to recreate call information from the NAS 120 when requests for that call are received.

For example, the NAS 120 authorized a call with the primary or local RPMS 124 that then lost network connectivity. The NAS 120 will attempt to send accounting updates to the primary or local RPMS 124 and fail, then roll over to the secondary or backup RPMS 136. If in the meantime the primary or local RPMS 124 comes back on line and the ISP customer 116 hangs up, the NAS 120 will send a resource-freed signal to the primary or local RPMS 124 which will then attempt to recreate the call context and free the necessary counters.

In the event that the secondary or backup RPMS 136 fails, it cannot be updated on the current counts. During this period and until all calls are closed that were active at the time it went down, the counts may be underreported. While in this state, the potential for over-subscription exists. When the secondary or backup RPMS 136 is brought back online, all counts from each primary or local RPMS 124 are transmitted to the secondary or backup RPMS 136 where they are aggregated to reflect counts for the entire network server group 112.

An example of a comma-delimited string representing the SSA can be seen in FIG. 4 in which the data from the string can be used to reconstruct the call information needed to maintain the connection through a RPMS. The example SSA and its syntax as an attribute/value pair in the NAS can be seen in the data string 150. The attribute/value pair may conform to TACACS+ or RADIUS or other such internet authentication protocol well known to those skilled in the art. The left segment may be the name 152 included in the attribute, which is depicted with an identifier prefix of "`Name "rm-server-state"`" in this example, separated from the right segment by a delimiter 154 shown as an equal sign. The right segment is the value 156, which is further subdivided into alphanumeric parameters that may be called "tokens" and separated by commas 158.

The value 156 begins with a first DNIS address 160 represented by a seven-digit integer. The second entry is the call type 162, which here is identified as "`speech`" followed by the third entry of CLID 164 represented by a seven-digit integer. The fourth entry is the modem port 166 represented by a five-field colon-separated parameter. The fifth entry is the resource group name 168 depicted by a period. The sixth and seventh entries are the call count 170 and the overflow count 172, each represented by an integer. The eighth, ninth, tenth, eleventh and twelfth entries, each denoted by a period and separated by commas 158, represent the start time 174, reference number 176, service group name 178, modem-tx-speed 180 and modem-rx-speed 182, respectively. The thirteenth entry represents the VPDN group name 184.

The data contained in the SSA data string 150 include information that cannot be obtained from the NAS administration call-status request. Source code to read the call data into the SSA and reconstruct the call information from the contents of the SSA may be written in Java language, although other programming languages are readily available to persons having ordinary skill in the art. To construct the SSA data string 150, the call data may appended together, separating each segment or token by a delimiter character such as a comma 158. For reading the SSA data string 150, each token may be read sequentially as a character string with the end of the token being determined by the presence of a delimiter character. The character string thus read may be placed into a string buffer to be written onto its appropriate data-field.

When an interruption between the NAS and its primary server occurs, the backup server issues an active-call petition to the NAS for the SSA data string 150. Once received, the server-state reconstruction subroutine in the backup server parses the SSA data-string 150 into the appropriate data functionalities. The code may check for the number of tokens in the data string to verify its completeness. The code may then sequentially place the character string of the token into a string buffer and write that character string from the string buffer to the corresponding data-field. This process may continue until each data-field has received an appropriate character string, whether read from the SSA data string or assigned by a default mechanism.

Such a process is described in the process flow diagram of FIG. 5, in which the tokens comprising the SSA data string are sequentially read and written onto to appropriate data-fields that describe the call attributes. Upon receiving the SSA data string, the subroutine 186 in the code reads the data string at reference 188. Using the delimiters as markers, the data string is separated into tokens at reference 190, with the number of tokens counted. The token number may be compared to the number expected n_{exp} as a read-error-detection tool at reference 192. If the number of tokens read is not equal to the number expected n_{exp}, the subroutine may halt for further instructions at reference 194. Assuming the two numbers are equal, a sequencing "loop" may be initiated at reference 196 with an example starting value of zero for the loop index i. The loop is then sequenced in sequencer 198 with the character string in the token placed in the buffer at reference 200. The loop index i may then be compared to a task value n, for a series of tasks $n = 0, 1, 2, \ldots, n_{\text{exp}}-1$ at reference 202 with the process branching or diverting accordingly to the appropriate data-assigning task that corresponds to the value of the loop index ($i = n$). Once the task associated with the loop index equaling the task value ($i = n$) has been completed, the loop index may be incremented at reference 204, in this example by one ($i = i+1$), and the loop sequencing process continued to compare the loop index with the subsequent task value.

Each task to write the contents of the buffer to a data-field corresponds to a particular task value, and the process branches to the task when the loop index equals the corresponding task value. If the loop index equals zero ($i = 0$) the DNIS may be extracted from the buffer and written to the DNIS data-field at reference 206a. If the loop index equals one ($i = 1$), the call type may be extracted from the buffer and written to the call type data-field at reference 206b. If the loop index equals two ($i = 2$), the CLID may be extracted from the buffer and written to the CLID data-field at reference 206c, and so forth shown as continuing steps 206d. If the loop index reaches one less than the number expected ($i = n_{\text{exp}}-1$), the VPDN may be extracted from the buffer and written to the VPDN data-field at reference 206e. Once completed, the incremented loop index equals the number expected ($i = n_{\text{exp}}$) and the completed condition is assigned the logic value of TRUE at reference 208, whereas the default prior to this would be FALSE. Once the data-fields are completed, the attribute information for the call is available and the call may be resumed by the server at reference 210.

The overall process for a backup server responding to a local server failure is summarized in the process flow diagram of FIG. 6. The network roll-over process may begin with a call being received by the NAS from a user at reference 212. The call is processed by the primary server at reference 214, wherein the server's function is performed by a primary RPMS. The call attributes are coded in a SSA, which is created and updated for characterizing the call at reference 216. This function is typically performed by the RPMS. The SSA may be included with each message sent to the NAS from the RPMS at reference 218. The NAS then receives and stores the SSA in memory at reference 220 for subsequent retrieval if needed.

A detector for alerting the NAS of a server failure may be employed at reference 222 to determine if the primary RPMS is still linked to the network. In the event that no failure has occurred, indicating that the primary RPMS is still functioning, the call continues to be processed by the primary RPMS at reference 214. However, if such a failure has occurred, this condition may be detected at reference 224, and the NAS responds by sending a message to the secondary server, in this case a backup RPMS, to continue processing the call at reference 226. The backup RPMS checks its database to determine if the call attribute information is available at reference 228. If not, the backup RPMS requests the NAS to provide the SSA at reference 230. The NAS sends the SSA to the backup RPMS at reference 232, and the backup RPMS then parses the SSA at reference 234 to obtain the latest call attribute information it requires. The backup RPMS then proceeds to process the call at reference 236. If the check by the backup RPMS for call attribute information determines the information's availability at reference 228, then the backup RPMS need not query the NAS and may proceed directly to process the call at reference 236. The steps 216, 218 and 220 provided through the primary RPMS processing of the call at reference 214 may also be continued by the backup RPMS processing of the call at reference 236.

While embodiments and applications of the invention have been shown and described, it would be apparent to those of ordinary skill in the art having the benefit of this disclosure, that many more modifications than mentioned above are possible without departing from the inventive concepts herein. The invention, therefore, is not to be restricted except in the spirit of the appended claims.